

NNN		NNN	MMM	MMM	LLL
NNN		NNN	MMM	MMM	LLL
NNN		NNN	MMM	MMM	LLL
NNN		NNN	MMMMMM	MMMMMM	LLL
NNN		NNN	MMMMMM	MMMMMM	LLL
NNN		NNN	MMMMMM	MMMMMM	LLL
NNNNNN		NNN	MMM	MMM	LLL
NNNNNN		NNN	MMM	MMM	LLL
NNNNNN		NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN		NNNNNN	MMM	MMM	LLL
NNN		NNNNNN	MMM	MMM	LLL
NNN		NNNNNN	MMM	MMM	LLL
NNN		NNN	MMM	MMM	LLL
NNN		NNN	MMM	MMM	LLL
NNN		NNN	MMM	MMM	LLL
NNN		NNN	MMM	MMM	LLL
NNN		NNN	MMM	MMM	LLLLLLLLLLLLLLLL
NNN		NNN	MMM	MMM	LLLLLLLLLLLLLLLL
NNN		NNN	MMM	MMM	LLLLLLLLLLLLLLLL

_S

Ps

NP

NP

SG

SO

NP

PA

_L

NN		NN	MM	MM	LL	DDDDDDDD	DDDDDDDD	LL	
NN		NN	MM	MM	LL	DDDDDDDD	DDDDDDDD	LL	
NN		NN	MMM	MMM	LL	DD	DD	LL	
NN		NN	MMM	MMM	LL	DD	DD	LL	
NNNN		NN	MM	MM	LL	DD	DD	LL	
NNNN		NN	MM	MM	LL	DD	DD	LL	
NN	NN	NN	MM	MM	LL	DD	DD	LL	
NN	NN	NN	MM	MM	LL	DD	DD	LL	
NN	NNNN	NN	MM	MM	LL	DD	DD	LL	
NN	NNNN	NN	MM	MM	LL	DD	DD	LL	
NN	NN	NN	MM	MM	LL	DD	DD	LL	
NN	NN	NN	MM	MM	LL	DD	DD	LL	
NN	NN	NN	MM	MM	LL	DD	DD	LL	
NN	NN	NN	MM	MM	LLLLLLLLLL	DDDDDDDD	DDDDDDDD	LLLLLLLLLL
NN	NN	NN	MM	MM	LLLLLLLLLL	DDDDDDDD	DDDDDDDD	LLLLLLLLLL

00000000		333333		222222
00000000		333333		222222
00	00	33	33	22
00	00	33	33	22
00	00		33	22
00	00		33	22
00000000		33		22
00000000		33		22
00	00		33	22
00	00		33	22
00	00		33	22
00	00	33	33	22
00	00	33	33	22
00000000		333333		2222222222
00000000		333333		2222222222

XTITLE 'NMLDDL - NML Data Definitions'
IDENT = 'V04-000'

```
*****
*
* COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
* DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
* ALL RIGHTS RESERVED.
*
* THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
* ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
* INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
* COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
* OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
* TRANSFERRED.
*
* THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
* AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
* CORPORATION.
*
* DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
* SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*
*****
```

++
FACILITY: DECnet-VAX Network Management Listener

ABSTRACT:

This module contains macro and symbol definitions used by all NML modules.

ENVIRONMENT: VAX/VMS Operating System

AUTHOR: Distributed Systems Software Engineering

CREATION DATE: 30-DEC-1979

MODIFIED BY:

V03-006	MKP0008	Kathy Perko	24-June-1984
		Increase the size of the QIO P4 buffer to the minimum value SYSGEN allows for MAX BUFFER. This is a slight improvement on the limit for the number of sources which can be logged for a single sink node.	
V03-005	MKP0007	Kathy Perko	9-April-1983
		Add globals for executor address in the volatile and permanent databases.	
V03-004	MKP0006	Kathy Perko	19-Sept-1983
		Convert node permanent database to multiple ISAM keys for better performance. Also, make NCP response message entity buffer	

size bigger and global - for X25 tracepoint names.

V03-003 MKP0005 Kathy Perko 19-April-1983
Delete service functions from NML.

V03-002 MKP0004 Kathy Perko 28-June-1982
Shrink P4 buffer size from 730 bytes to 512 to get rid
of QIO quota exceeded errors.
Add macro for generating Search Key IDs for Entity Table.
Rename qualifier to use its CPT index instead of the Network
Management parameter code.

V03-001 MKP0003 Kathy Perko 17-Mar-1982
Rename some global fields so the names mean more.

V02-002 MKP0002 Kathy Perko 2-Nov-1981
Delete NML\$GW_CMD_CHAN

V02-001 MKP0001 Kathy Perko 14-Sept-1981
Make P4 buffer size smaller so systems with SYSGEN parameter,
MAXBUF, don't get buffer quota exceeded for SHOW CIRCUIT
CHARACTERISTICS.

Miscellaneous symbols

LITERAL

FALSE = 0,
TRUE = 1;

The following symbols are internal parameter codes. The values all have
bit 15 set, indicating a counter value, to avoid conflicts with other
network management parameter codes.

LITERAL

NMASC_PCNO_ASS = 1 ^ 15 OR 0,	Loop node address
NMASC_PCLI_LCS = 1 ^ 15 OR 1,	Line counters
NMASC_PCNO_ECS = 1 ^ 15 OR 2,	Executor node counters
NMASC_PCNO_NCS = 1 ^ 15 OR 3,	Node counters
NMASC_PCCI_CCS = 1 ^ 15 OR 4,	Circuit counters
NMASC_PCXP_PCS = 1 ^ 15 OR 5,	X-25 Protocol DTE counters.
NMASC_PCXS_SCS = 1 ^ 15 OR 6;	X-25 Server counters

Structure declarations used for system defined structures to
save typing.

STRUCTURE

BBLOCK [O, P, S, E; N] =
[N]
(BBLOCK+O)<P,S,E>,

BBLOCKVECTOR [I, O, P, S, E; N, BS] =
[N*BS]
((BBLOCKVECTOR+I*BS)+O)<P,S,E>;

Macro to generate Network ACP Control QIO (NFB) P1 buffer contents. The NFB describes SET, SHOW, CLEAR, and ZERO operations.

MACRO

```

$NFB (FUNC, FLAGS, DATABASE, SRCH_KEY_ONE, OPER_ONE,
      SRCH_KEY_TWO, OPER_TWO) =
  BYTE ( %IF %IDENTICAL (FUNC, 0)           ! QIO function code.
        %THEN 0
        %ELSE %NAME ('NFB$C_FC_',FUNC)
        %FI),
  BYTE ( %IF %NULL (FLAGS)                  ! Error Update and Process
        %THEN 0                             ! Multiple Entries flags.
        %ELSE FLAGS
        %FI),
  BYTE ( %IF %IDENTICAL (DATABASE, 0)        ! ACP database to update.
        %THEN 0
        %ELSE %NAME ('NFB$C_DB_',DATABASE)
        %FI),
  BYTE (%IF %NULL (OPER_ONE)                 ! Oper1
        %THEN 0
        %ELSE OPER_ONE
        %FI),
  $SRCH_KEY (DATABASE, SRCH_KEY_ONE),        ! Search key one ID
  $SRCH_KEY (DATABASE, SRCH_KEY_TWO),        ! Search key two ID
  BYTE (%IF %NULL (OPER_TWO)                 ! Oper2
        %THEN 0
        %ELSE OPER_TWO
        %FI),
  BYTE (0),                                 ! Spare
  WORD (0),                                 ! variable cell size

  %IF NOT %NULL(%REMAINING)
  %THEN $FIELD_ID_LIST (DATABASE, %REMAINING)
  ,LONG (NFB$C_ENDOFLIST) ! End delimiter for field ID list.
  %ELSE
  LONG (NFB$C_ENDOFLIST) ! End delimiter for field ID list.
  %FI
%,

```

Generate a Search Key ID for an NFB. If the Search key is null, use a wildcard search key ID.

```

$SRCH_KEY (DATABASE, SRCH_ID) =
  LONG ( %IF %NULL (SRCH_ID)
        %THEN NFB$C_WILDCARD
        %ELSE $FIELD_ID (DATABASE, SRCH_ID)
        %FI )
%,

```

```
! Generate a list of longwords containing the NETACP field IDs for
! the parameters. This iterative macro will generate as many
! field IDs as are supplied.
```

```
$FIELD_ID_LIST (DATABASE) [FIELD_ID] =
  LONG T$FIELD_ID (DATABASE, FIELD_ID))
  %;
```

```
$FIELD_ID (DATABASE, FIELD_ID) =
  %IF %IDENTICAL (FIELD_ID, NFB$C_WILDCARD) OR
  %IDENTICAL (FIELD_ID, NFB$C_COLLATE)
  %THEN
    FIELD_ID
  %ELSE
    %IF %NULL (FIELD_ID)
    %THEN 0
    %ELSE %NAME ('NFB$C_', DATABASE, '_', FIELD_ID)
    %FI
  %FI
  %;
```

```
! Macros to generate Network Control I/O request descriptors.
```

```
MACRO
```

```
! Declare the NFB buffer (use the number of input parameters to figure
! out how big to make it) and set up a descriptor for it.
```

```
$NFB$DSC (NAM) =
  SWITCHES UNAMES;
  OWN
    _NFB : VECTOR [$NFB_ALLOCATION (%REMAINING)]
    INITIAL ($NFB (%REMAINING));
  BIND
    %NAME (NAM) = UPLIT (%ALLOCATION(_NFB), _NFB);
  UNDECLARE _NFB;
  SWITCHES NOUNAMES
  %;
```

```
$NFB_ALLOCATION [] =
  5+(MAX(0,%LENGTH-6))
  %;
```

```
! Macro to extract the bit number from bit field references
```

```
MACRO
```

```
$BITN (O, B, W, S) = B
  %;
```

```
! Macro to signal status message
```

```
MACRO
```

```
$SIGNAL MSG [] =
  SIGNAL (NML$K_SIG_CODE, %REMAINING)
  %;
```

Macro to create constant string descriptor

```
MACRO
$ASCID [] =
    (UPLIT (%CHARCOUNT(%STRING(%REMAINING)),
    UPLIT BYTE (%STRING(%REMAINING)))
%;
```

```
MACRO
$ASCIC [] =
    UPLIT BYTE (%ASCIC %STRING (%REMAINING))
%;
```

Macro to move an ASCII counted string to a buffer.

```
MACRO
$MOVE_ASCIC (STRING, PTR) =
    PTR = CH$MOVE ( %CHARCOUNT (%ASCIC STRING),
    UPLIT BYTE (%ASCIC STRING),
    .PTR)
%;
```

```
MACRO
DESCRIPTOR =
    BBLOCK [8]
%;
```

I/O Status Block definition

```
FIELD
IOSB_FIELDS =
    SET
    IOS$W_STATUS = [0, 0, 16, 0],    ! Status field
    IOS$W_COUNT = [2, 0, 16, 0],    ! Byte count field
    IOS$L_INFO = [4, 0, 32, 0]    ! Device dependent information
    TES;
```

```
MACRO
$IOSB =
    BBLOCK [8] FIELD (IOSB_FIELDS)
%;
```

Macro to define Network Management version fields

```
FIELD
NMV_FIELDS =
    SET
    NMV$B_VERSION = [0,0,8,0],
    NMV$B_DEC_ECO = [1,0,8,0],
    NMV$B_USER_ECO = [2,0,8,0]
    TES;
```

```
MACRO
NMV = BBLOCK [3] FIELD (NMV_FIELDS)
```

%:

Macro to define external symbols common to most of the modules.

MACRO \$NML_EXTDEF =
EXTERNAL

Event data

NML\$GB_EVTSRCTYP	: BYTE,	! Event source type
NML\$GQ_EVTSRCDS	: DESCRIPTOR,	! Event source descriptor
NML\$GW_EVTCLASS	: WORD,	! Event class
NML\$GB_EVTMSKTYP	: BYTE,	! Mask type
NML\$GQ_EVTMSKDS	: DESCRIPTOR,	! Mask descriptor
NML\$GW_EVTSNKADR	: WORD,	! Sink node address

NML\$GW_ACP_CHAN,	
NML\$GL_LOGMASK	: BITVECTOR [32],
NML\$GQ_ENTSTRDSC	: DESCRIPTOR,
NML\$AB_QIOBUFFER	: BBLOCK [0],
NML\$GQ_QIOBFDSC	: DESCRIPTOR,
NML\$AB_EXEBUFFER	: VECTOR [0, BYTE],
NML\$GL_EXEDATPTR,	
NML\$GQ_EXEDATDSC	: DESCRIPTOR,
NML\$GQ_EXEBFDSC	: DESCRIPTOR,
NML\$AB_RCVBUFFER	: VECTOR [NML\$K_RCVBFLEN, BYTE],
NML\$GQ_RCVBFDSC	: DESCRIPTOR,
NML\$AB_SNDBUFFER	: VECTOR [NML\$K_SNDBFLEN, BYTE],
NML\$GQ_SNDBFDSC	: DESCRIPTOR,
NML\$GL_RCVDATLEN,	
NML\$AB_CPTABLE	: BBLOCKVECTOR [0, CPT\$K_ENTRYLEN],
NML\$AB_MSGBLOCK	: BBLOCK [MSB\$K_LENGTH],
NML\$AB_ENTITY_ID	: BBLOCK [16],
NML\$AB_QUALIFIER_ID	: BBLOCK [16],
NML\$AB_ENTITYDATA	: BBLOCKVECTOR [, EIT\$K_ENTRYLEN],
NML\$AB_NML_NMV	: NMV,
NML\$AB_PRMSEM	: BBLOCKVECTOR [0, PST\$K_ENTRYLEN],
NML\$AB_RECBUF	: BBLOCK [0],
NML\$AL_ENTINF TAB	: VECTOR [0],
NML\$AL_PERMINTAB	: VECTOR [0],
NML\$AW_PRM_DES	: BLOCKVECTOR [PDB\$K_NUMBER, 4, WORD],
NML\$GB_CMD_VER	: BYTE,
NML\$GB_ENTITY_CODE	: BYTE,
NML\$GB_ENTITY_FORMAT	: BYTE,
NML\$GL_QUALIFIER_PST,	
NML\$GB_QUALIFIER_FORMAT	: BYTE,
NML\$GB_FUNCTION	: BYTE,
NML\$GB_INFO	: BYTE,
NML\$GB_OPTIONS	: BYTE,
NML\$GL_PRM CODE,	
NML\$GL_PRS_FLGS	: BLOCK [1],
NML\$GL_NML_ENTITY,	
NML\$GQ_NETRANDSC	: DESCRIPTOR,
NML\$GQ_RECBFDSC	: DESCRIPTOR,
NML\$GW_PRMDESCNT	: WORD;

%:

```

: NPARSE argument block structure definitions

```

```

MACRO
  $NPA_ARGDEF =
    BUILTIN
    AP;
  BIND
    NPARSE_BLOCK = AP : REF $NPA_BLKDEF;
  %;

```

```

: NPARSE argument block definition macro

```

```

MACRO
  $NPA_BLKDEF =
    BBLOCK [NPASK_LENGTH0]
  %;

```

```

: Buffer length parameters.

```

```

LITERAL
  NML$K_RCVBFLEN = 512,      ! Receive buffer length
  NML$K_SNDBFLEN = 512,      ! Send buffer length
  NML$K_NFBBFLEN = 256,      ! Max size for an NFB.
  NML$K_QIOBFLEN = 1200,     ! QIO buffer length
  NML$K_P2BUFLEN = 104,      ! Max length for P2 buffers.
  NML$K_RECBFLEN = 1024,     ! Record buffer length
  NML$K_ENTBUFLEN = 64,      ! Entity name buffer size.

  ! Maximum bytes of data in a permanent database record. Leaves room
  ! for the node keys (which take up the most room) at the beginning of
  ! the record.
  NML$K_MAX_REC_DATA = NML$K_RECBFLEN - NMN$K_NODE_KEYS_LEN,
  NML$K_PERM_KEYS_LEN = 2;    ! Key length for all permanent databases
                              ! except the node database.

```

```

: Parameter descriptor block (PDB) definitions.

```

```

LITERAL PDB$K_NUMBER      = 32;      ! Number of parameter descriptor slots

MACRO   PDB$W_INDEX        = 0,0,16,0%; ! Parameter change table (CPT) index
MACRO   PDB$W_COUNT        = 1,0,16,0%; ! Parameter byte count
MACRO   PDB$A_POINTER      = 2,0,32,0%; ! Pointer to parameter value

LITERAL PDB$K_SIZE        = 8;      ! Size of parameter descriptor entry

```

0280 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY